Postel

Network Working Group — M. Elie

Request for Comments #64 — UCLA

Getting Rid of Marking

Though we realize that this improvement is perhaps somewhat late to be implemented, we believe that there exist better solutions than marking and suggest a simple modification to the IMP-HOST interface which would avoid it.

1. The harm.

Marking was introduced to suit the sending Host because it permits the text of a message to start on a word boundary; however, it does not suit the receiving Host with a different word length. Moreover, it introduces in the message useless bits. Let us illustrate this by the example of our Sigma 7, a 32 bits machine.

1.1 Inefficiency in Computation

Suppose we receive a message from an 18 bit machine (figure 1.1) coded in 8 bit ASCII characters which will eventually become standard on the network. In order to translate this message into our EBCDIC internal code, for instance,

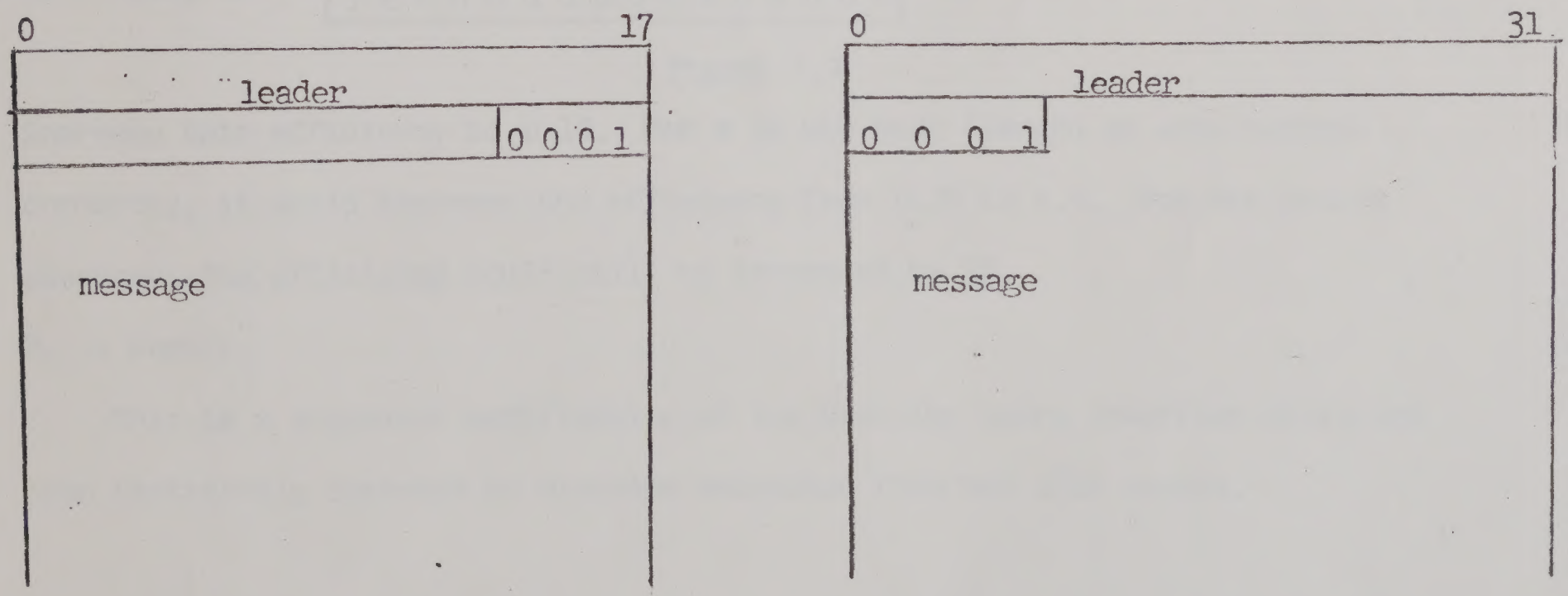


figure 1.1

we first have to shift the whole message. We must detect the first 1 following the leader, and from this determine that we must shift the message 4 bits to the left. This takes approximately 12 μsec per double word, which makes 1.5 msec per full regular message. This is not huge, but still it is about one-third of the time it will take to translate the message in internal code.

1.2 Inefficiency in transmission

More important is the inefficiency resulting from adding unnecessary bits to the message, especially if it turns out that one character messages are used. Figure 1.2 shows the example of a 1 character text sent by the sigma 7, which results in transmitting 112 bits to carry 8 bits of information, thus leading to an efficiency factor of 0.07. Suppression of marking would

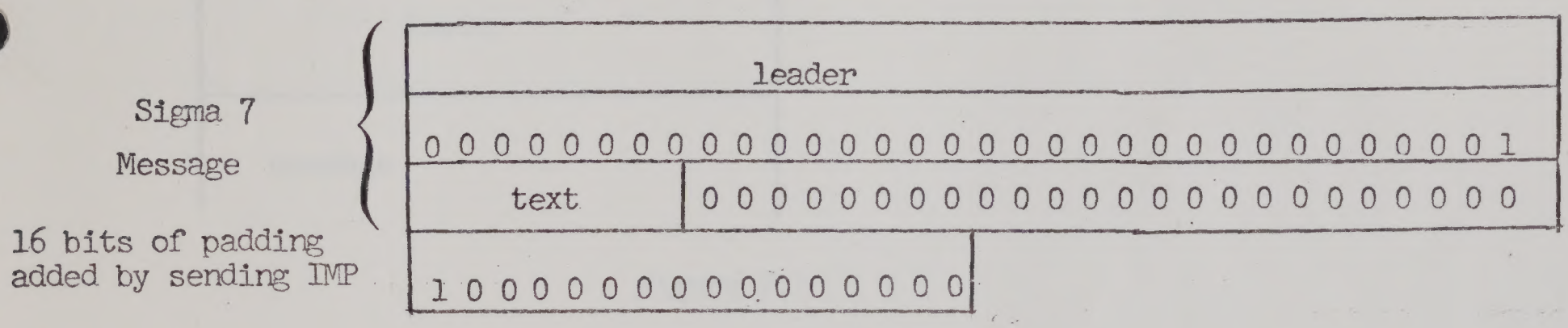


figure 1.2

increase this efficiency to 0.10. For a 32 bit text (length of some control commands), it would increase the efficiency from 0.28 to 0.4. For one packet messages, the efficiency would still be increased by 3%.

2. A remedy.

This is a suggested modification of the Host-Imp users interface which has been tentatively sketched on diagrams extracted from BBN 1822 report.

2.1 Host to Imp

The modification consists of adding a counter to 32, enabled at the beginning of a message, and incremented at each bit passed to the IMP; when it reaches 32 it forces a "word complete" signal asking for a new word in the shift register and resetting the word length counter; thus the unused bits in the last word of the leader are not transmitted and the message starts with the next word (see figure 2.1).

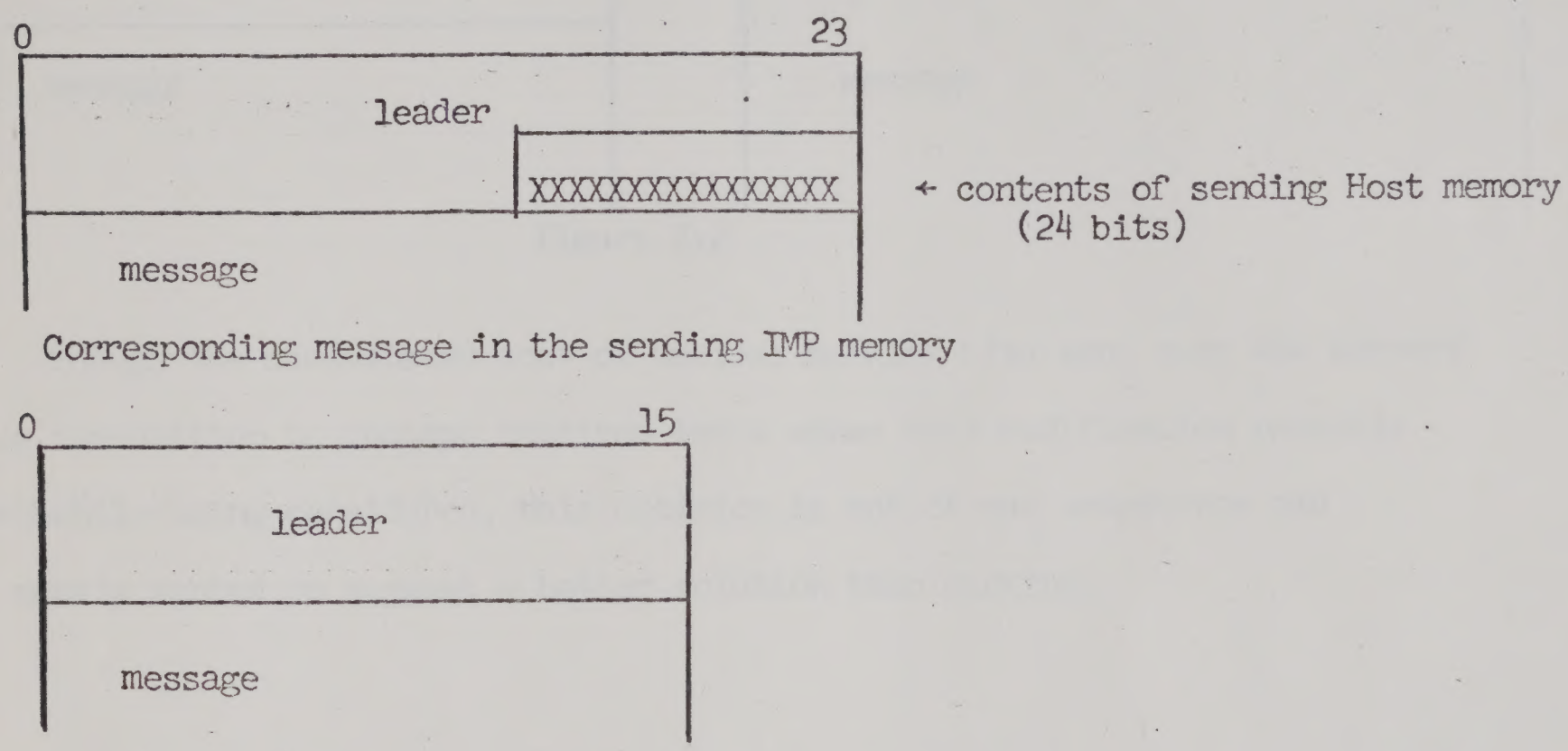


figure 2.1

2.2 Imp to Host

The modification consists of adding a counter to 32. When 32 bits have entered the shift register from the Imp at the beginning of a new message, the counter allows the register to be shifted up to the point to be full (which is detected by the word length counter) without entering any new bit from the Imp.

Thus, the next bit of the message which is the first bit of text, will be entered as the first bit of the next word (see figure 2.2).

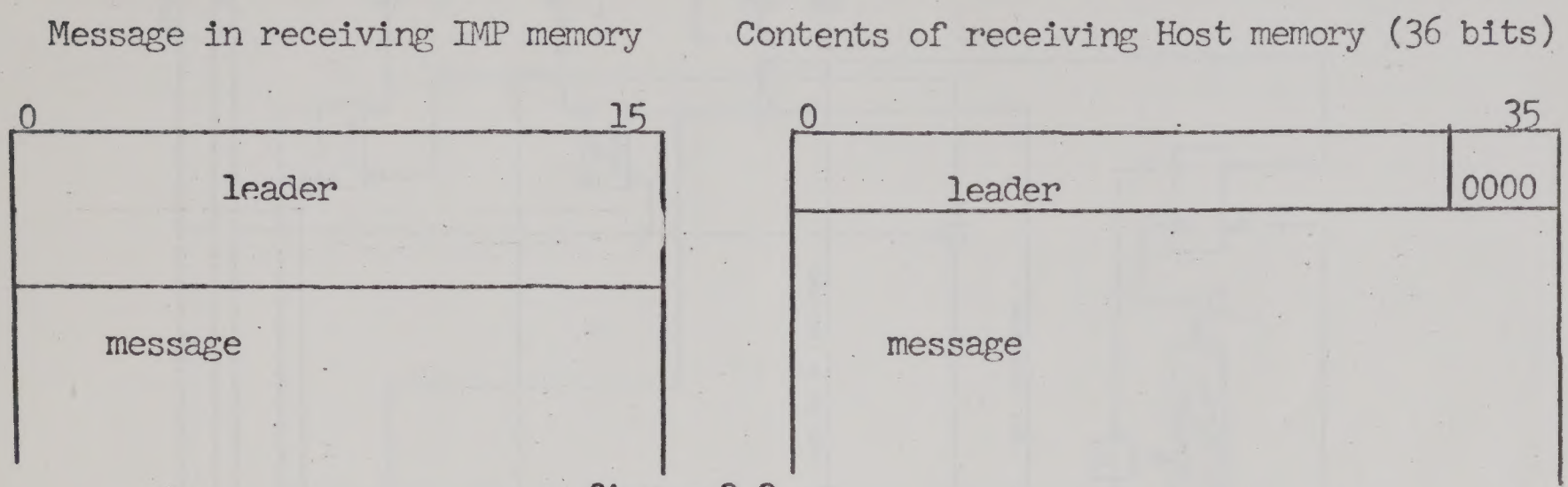


figure 2.2

Though the accumulated cost of useless marking bits sent over the network plus computation to reshape received texts makes this modification probably worthwhile being considered, this decision is not of our competence and we merely wanted to suggest a better solution than marking.

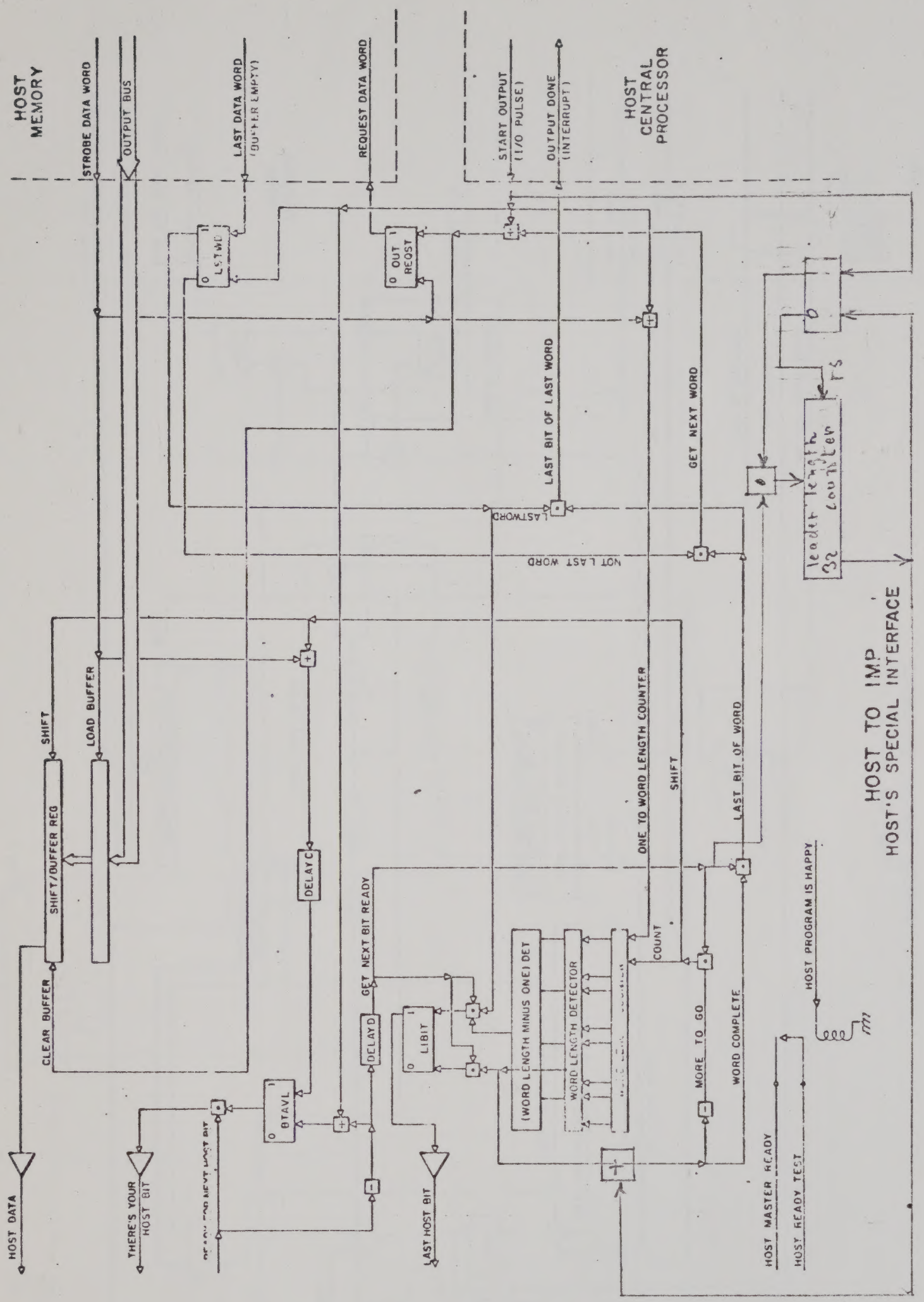
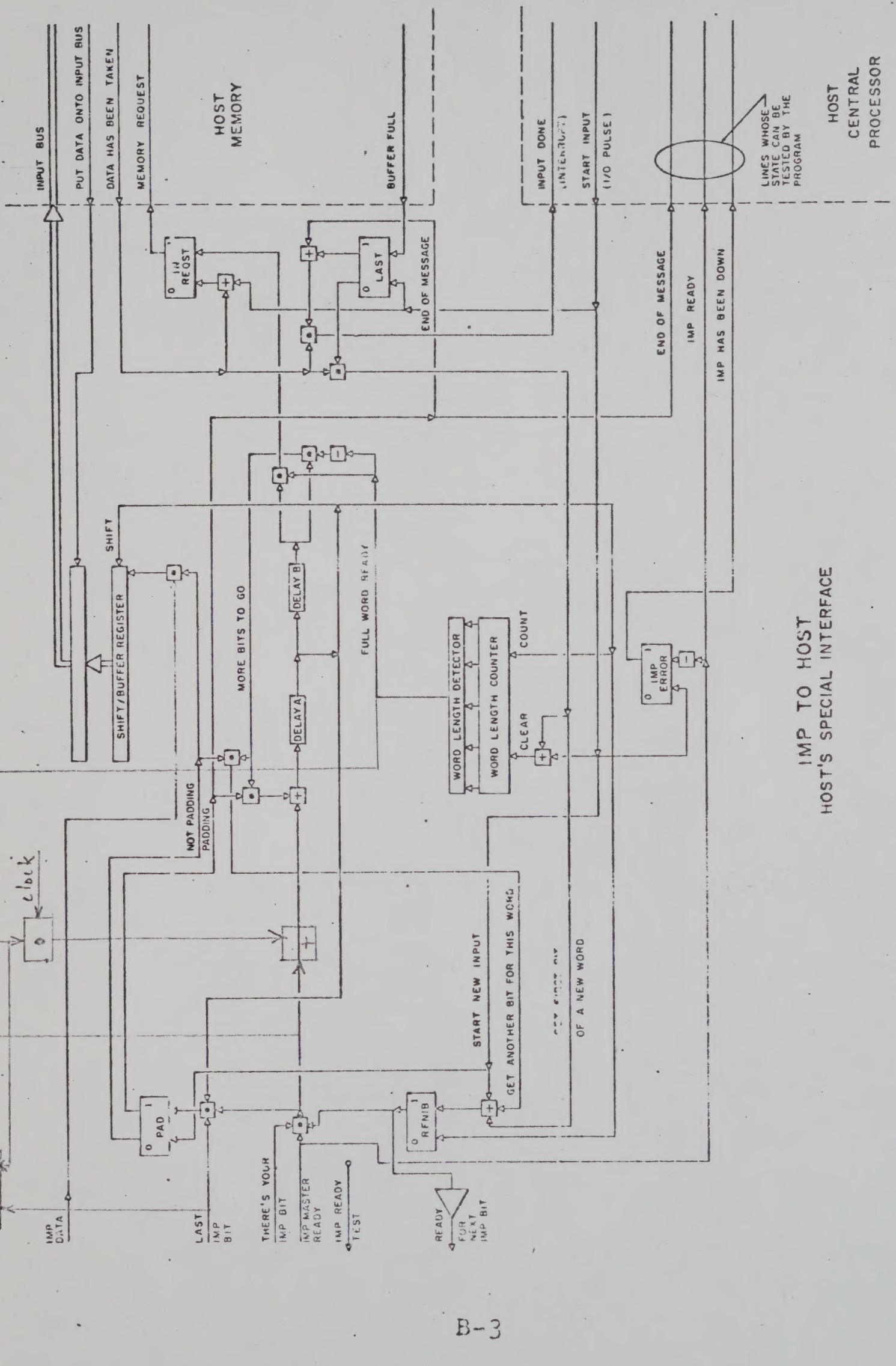

HOST MEMORY
STROBE DATA WORD
OUTPUT BUS
LAST DATA WORD
REQUEST DATA WORD
START OUTPUT (I/O PULSE)
OUTPUT DONE (INTERRUPT)
HOST CENTRAL PROCESSOR
LSTWD
OUT REQST
LAST BIT OF LAST WORD
GET NEXT WORD
LASTWORD
NOT LAST WORD
leader length 32 counter
SHIFT
LOAD BUFFER
SHIFT/BUFFER REG
CLEAR BUFFER
DELAYC
DELAYD
GET NEXT BIT READY
BTAVL
LIBIT
(WORD LENGTH MINUS ONE) DET
WORD LENGTH DETECTOR
ONE TO WORD LENGTH COUNTER
SHIFT
COUNT
LAST BIT OF WORD
MORE TO GO
WORD COMPLETE
HOST DATA
THERE'S YOUR HOST BIT
LAST HOST BIT
HOST MASTER READY
HOST READY TEST
HOST PROGRAM IS HAPPY
HOST TO IMP
HOST'S SPECIAL INTERFACE

FIGURE B-1



IMP TO HOST
HOST'S SPECIAL INTERFACE

FIGURE B-2